La fougère ou les L-systèmes*
2014-2015

Par Quentin MASSE, Robin WELYKYJ élèves de seconde et Tanguy RUTH, Martin LUC élèves de terminale S au Lycée d'Altitude de Briançon (Hautes-Alpes-05)
Par Victor Motogna, Meda Muresan, Mihai Chindris, classe XII, Alexandra Maior, Sara Boancă et Uliana Palade, classe XI, Maria Craciun, Iulia Rebreanu, Alexandra Timoce, classe X, Curta Antonia, Coruțiu Mara, Mîrza Lorena classe IX du Colégiul National Emil Racovita de Cluj (Roumanie)

Enseignantes : Valentina Vasilescu, Adrian-Vasile Andrea et Ariana-Stanca Vacaretu
Chercheuse : Adela Lupescu (Universitatea Bades-Bolyai de Cluj-Napoca)
Enseignants : Hubert PROAL et Mickaël LISSONDE
Chercheurs : Camille PETIT (Université de Fribourg) et Yves PAPEGAY (INRIA-Sophia Antipolis).
Vous pouvez télécharger des documents et simulations sur ce sujet sur le site
http://www.informathiques.fr/MeJ.html de Yves Papegay.

Présentation du sujet

Nous disposons d'un alphabet composé de deux lettres : B (bourgeon) et F (tige)
Nous définissons deux règles :
B → F[+B][-B]FB
F → FF
Objectif :
Étudier la suite (longueur, nombre de B et de F), déterminer la longueur de la branche centrale, réaliser un programme, changer l'alphabet et les règles....

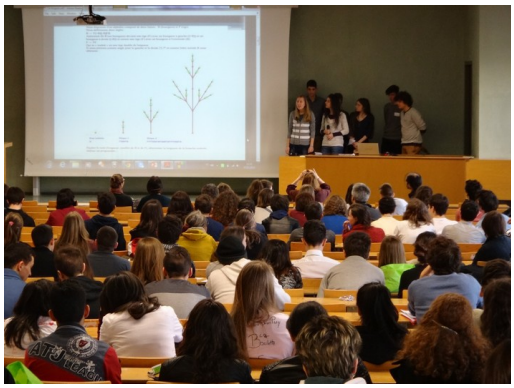
Résultats obtenus

Méthode pour calculer la somme de termes d'une suite particulière.

Valorisations des travaux

Présentations lors de la semaine des maths à Briançon (mars 2015)
Présentation lors du « Forum of the math research projects » à Cluj-Napoca (mars 2015)
Exposé commun lors du congrès *MATh.en.JEAN'S* à l'Université d'Avignon (mars 2015)
Présentation lors du « Math Youth Forum » à Cluj-Napoca (mai 2015)
Présentations lors du forum PASS à Aix-en-Provence (juin 2015)

Séminaire entre les deux établissements à Cluj



Exposé du sujet lors du congrès MeJ d'Avignon

Sujet :

Nous disposons d'un alphabet composé de deux lettres : B (bourgeon) et F (tige)

Nous définissons deux règles :

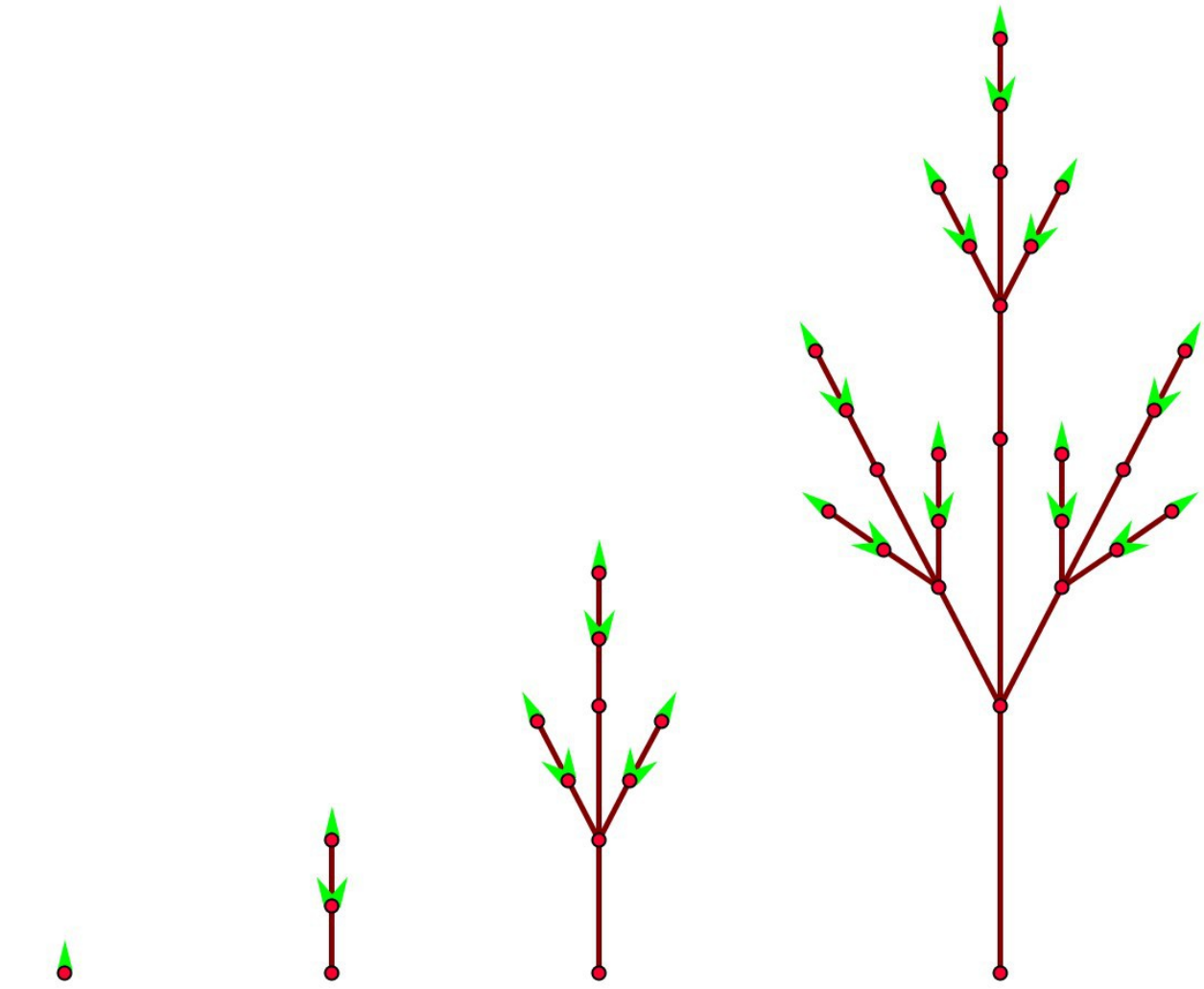
B → F[+B][-B]FB

Autrement dit B (un bourgeon) devient une tige (F) avec un bourgeon à gauche ([+B]) et un bourgeon à droite ([-B]) et encore une tige (F) avec un bourgeon à l'extrémité (B).

F → FF

Qui se « traduit » en une tige double de longueur.

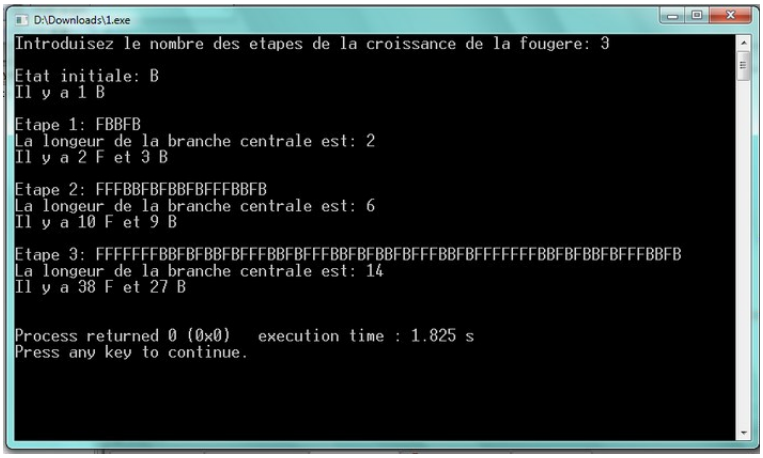
Si nous prenons comme angle pour la gauche et la droite 25,7° et comme lettre initiale B nous obtenons :



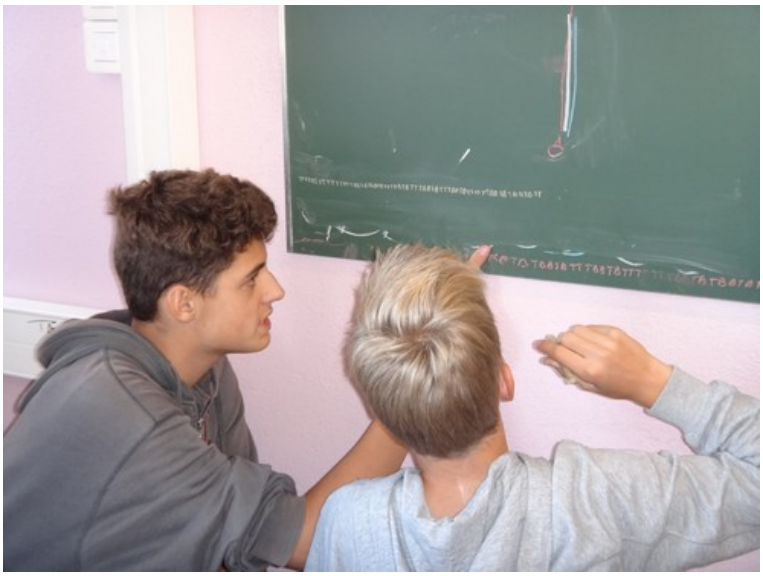
Étudier la suite (nombre de B et de F), déterminer la longueur de la branche centrale, réaliser un programme, changer l'alphabet et les règles....

I. Démarche expérimentale

Nous sommes partis dans plusieurs directions : compter le nombre de B et F « à la main », essayer de faire évoluer la liste en programmant sur un tableur, dessiner les différents états.
Suite à la première vidéo conférence qui nous a permis d'échanger avec les élèves de terminale et les élèves roumains, nous avons repris nos premiers résultats et essayé de trouver des formules qui correspondent. En effet les élèves roumains ont présenté un programme (voir annexe 1) qui détermine le nombre de B et de F en fonction de l'étape rentée au clavier.



Copie d'écran du programme des élèves de Cluj qui donne la séquence de F et de B en fonction du numéro de l'étape d'évolution

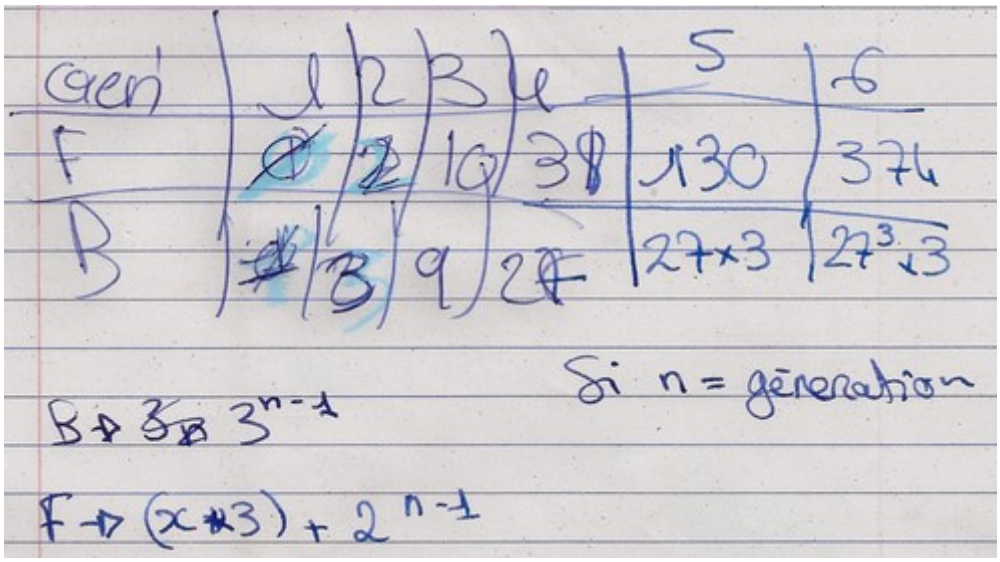


Recherche de la séquence de F et de B, "à la main" par les élèves de Briançon

Si l'on note le nombre de B et de F selon les états, nous obtenons deux suites* :

État	0	1	2	3	4
Nombre de F (F_n)	0	2	10	38	114
Nombre de B (B_n)	1	3	9	27	81

Avant d'introduire la notation sous forme de suite, nous écrivions des formules du type :



Extrait du cahier de recherche où l'écriture sous forme de suite n'est pas encore introduit

$2^{i-1}\times F+a+a+2^{i-1}\times F+a$ où a est la formule de la fougère de l'étape antérieure et i est le nombre de l'étape, autrement écrit cette formule devient $F_n=2^{n-1}+F_{n-1}+F_{n-1}+2^{n-1}+F_{n-1}=3F_{n-1}+2^n$ (3). Cette formule vient de l'observation suivante des élèves de seconde et des élèves roumains : pour obtenir l'image de la fougère de l'étape suivante il faut seulement ajouter l'image de la fougère de l'étape antérieure une fois à droite, une fois à gauche et une fois au-dessus de l'image qui représente la tige, comme nous l'avons représenté dans les illustrations 1.

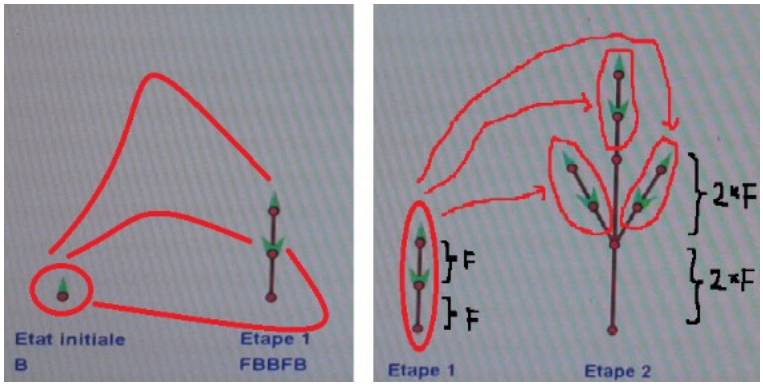


Illustration 1: Explication de l'élaboration de la formule de récurrence donnant le nombre de F

II. Analyse des résultats et propriétés

<u>Propriété 1</u> : La suite des B correspond aux puissances de 3, nous pouvons annoncer que $B_n=3^n$ (1)

Démonstration de la propriété 1 :
Si l'on regarde les deux règles, seule la règle B → F[+B][-B]FB fait apparaître des B et pour un B à l'état n nous avons 3B à l'état n+1. Ainsi la suite B_n est géométrique* de raison 3 et de premier terme $B_0=1$
Alors $B_n=B_0\times q^n=3^n$

Pour déterminer le nombre de F, nous sommes partis dans deux directions.

Démarche des élèves de terminale :
Nous nous sommes dit que tous les F à l'état n donneront 2F à l'état d'après (règle 2) et que chaque B fournira 2F (règle 2). Ainsi $F_{n+1}=2F_n+2B_n=2F_n+2\times3^n$ avec $F_0=0$ (2)
Pour obtenir une formule pour F_n en fonction de n, nous avons procédé ainsi :
$F_{n+1}=2F_n+2\times3^n$
$F_{n+1}=2(2F_{n-1}+2\times3^{n-1})+2\times3^n=2^2F_{n-1}+2^2\times3^{n-1}+2\times3^n$
$F_{n+1}=2^2(2F_{n-2}+2\times3^{n-2})+2^2\times3^{n-1}+2\times3^n=2^3F_{n-2}+2^3\times3^{n-2}+2^2\times3^{n-1}+2\times3^n$

<u>Propriété 2</u> : $F_n=\sum_{k=1}^{n}2^k\times3^{n-k}=2\times\sum_{k=0}^{n-1}2^k\times3^{n-1-k}$ pour $n\geqslant1$ et $F_0=0$

Démonstration par récurrence* de la propriété 2 :
Initialisation : $n=1$
$F_1=2$ et $\sum_{k=1}^{1}2^k\times3^{1-k}=2^1\times3^0=2$ Donc la propriété est vérifiée pour $n=1$.

Hérédité : On suppose qu'il existe un n tel que $F_n=\sum_{k=1}^{n}2^k\times3^{n-k}$, on veut montrer que

$F_{n+1}=\sum_{k=1}^{n+1}2^k\times3^{n+1-k}$

Nous partons de l'hypothèse de récurrence $F_n=\sum_{k=1}^{n}2^k\times3^{n-k}$ que nous multiplions par 2 et nous additionnons 2×3^n. Nous obtenons

$F_{n+1}=2\times\left(\sum_{k=1}^{n}2^k\times3^{n-k}\right)+2\times3^n=\sum_{k=1}^{n}2^{k+1}\times3^{n-k}+2\times3^n=\sum_{k=1}^{n+1}2^k\times3^{n+1-k}$

Conclusion : la propriété a été démontrée par récurrence.

Remarque : deux groupes ont obtenu deux résultats différents (2) et (3) pour la formule de F_n qui vont conduire à la même formule finale.

La formule de la propriété 2 ne nous convenait pas, nous avons cherché à la simplifier.
Avec l'aide de notre chercheur nous avons la propriété suivante :

<u>Propriété 3</u> : $F_n=2(3^n-2^n)$ pour $n\geqslant0$.

Démonstration par récurrence de la propriété 3. Voir l'annexe 2.

Lors du séminaire avec les élèves de Cluj, ces derniers ont proposé une méthode plus élégante pour passer de la formule de la propriété 2 à celle de la propriété 3.
Voici leur démarche :
$F_n=\sum_{k=1}^{n}2^k\times3^{n-k}$ si on multiplie par 3^{-n} nous avons

$$3^{-n}\times F_n=\sum_{k=1}^{n}2^k\times3^{-k}=\sum_{k=1}^{n}\left(\frac{2}{3}\right)^k=\left(\frac{2}{3}\right)\frac{\left(\frac{2}{3}\right)^n-1}{\frac{2}{3}-1}=2\left(1-\left(\frac{2}{3}\right)^n\right)=2\times3^{-n}(3^n-2^n)$$

ce qui nous donne $F_n=2(3^n-2^n)$

Pour être exacte, voici le raisonnement des élèves de Cluj :

Cette forme de $F_n=\sum_{k=1}^{n} 2^k\times 3^{n-k}$ est la somme des termes d'une progression géométrique de raison $\frac{2}{3}$.

En utilisant la formule de la somme des termes d'une progression géométrique on obtient :

$F_n=2\left(3^n-2^n\right)$

Nous avons ensuite cherché la « hauteur » le la fougère, c'est-à-dire la longueur de la tige centrale ou encore le nombre de F sur la tige centrale.

Nous notons L_n le nombre de F sur la tige centrale à l'étape *n*.

Vous pouvez voir le résultat obtenu et le programme en C++ des élèves roumains en annexe 3.

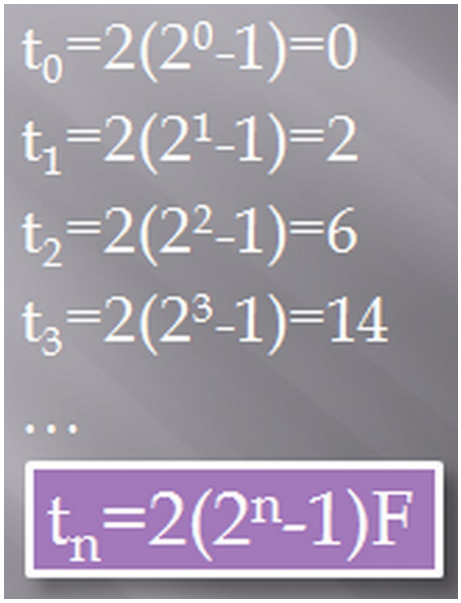
Propriété 4 : la longueur de la fougère à l'étape *n* est donnée par $L_n=2L_{n-1}+2$ avec $L_0=0$

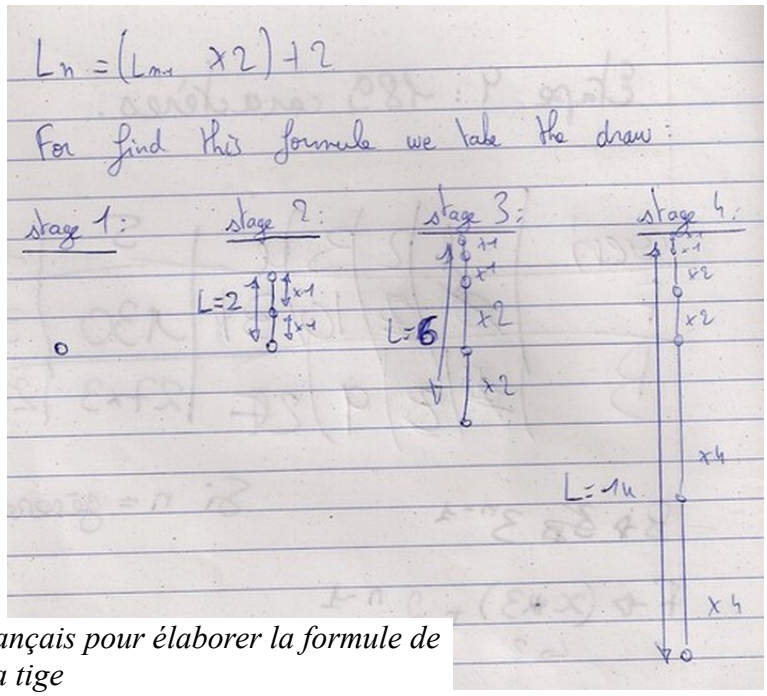
Démonstration de la propriété 4 :

Si nous regardons la tige centrale à l'étape *n*, le nombre de F va donner la longueur, elle est composée de L_{n-1} F qui a l'étape suivante vont doubler et d'un B (sans longueur) qui va donner FFB (sur sa partie centrale). Ainsi le nombre de F sera de $2L_{n-1}+2$

Étape	0	1	2	3	4	5
L	0	2	6	14	30	62

Propriété 5 : après plusieurs essais, les élèves des deux établissements ont obtenu la formule $L_n=2^{n+1}-2=2\left(2^n-1\right)$ pour $n\geqslant 0$. (4)





Extraits des travaux des élèves roumains et français pour élaborer la formule de la longueur de la tige

Démonstration de la propriété 5 :

La longueur de la fougère à l'étape *n* est donnée par $L_n=2L_{n-1}+2$ avec $L_0=0$

Si on considère la suite $V_n=L_n+2$, on peut montrer que la suite V_n est géométrique :

$V_{n+1}=L_{n+1}+2=2L_n+2+2=2(L_n+2)=2V_n$ donc V_n est géométrique de raison 2 et de premier terme

$V_0=2$

Nous obtenons alors $V_n=V_0\times q^n=2\times 2^n=2^{n+1}$ et $L_n=V_n-2=2^{n+1}-2$ pour $n\geqslant 0$

Autre démonstration de la propriété 5 par des élèves roumains :

$L_{n+1}=2L_n+2$

$L_n=2L_{n-1}+2 \quad \times 2$

$L_{n-1}=2L_{n-2}+2 \quad \times 2^2$

…

$L_2=2L_1+2 \quad \times 2^{n-1}$

$L_1=2L_0+2 \quad \times 2^n$

Si nous faisons la somme des égalités, des termes vont se simplifier.

$L_{n+1}+2L_n+2^2L_{n-1}+\ldots+2^{n-1}L_2+2^nL_1=2L_n+2\times 2L_{n-1}+2^2\times 2L_{n-2}+\ldots+2^nL_1+2^{n+1}L_0+2+2^2+2^3+\ldots+2^n+2^{n+1}$

Mais au début (à l'étape 0) nous n'avons aucune tige, $L_0=0$

$L_{n+1}=\sum_{k=1}^{n+1}2^k=2^{n+2}-2$ autrement dit $L_n=2^{n+1}-2$

III. Autre modèle

Nous avons l'impression que le doublement de la tige (règle F → FF) fait grandir trop vite le modèle.
Nous avons modifié l'alphabet en rajoutant une lettre G. Graphiquement un G est comme un F c'est à dire un vecteur de longueur 1 (illustration 2).
La règle 1 devient B → G[+B][-B]GB
La règle 2 devient G → FG et F n'évolue pas.

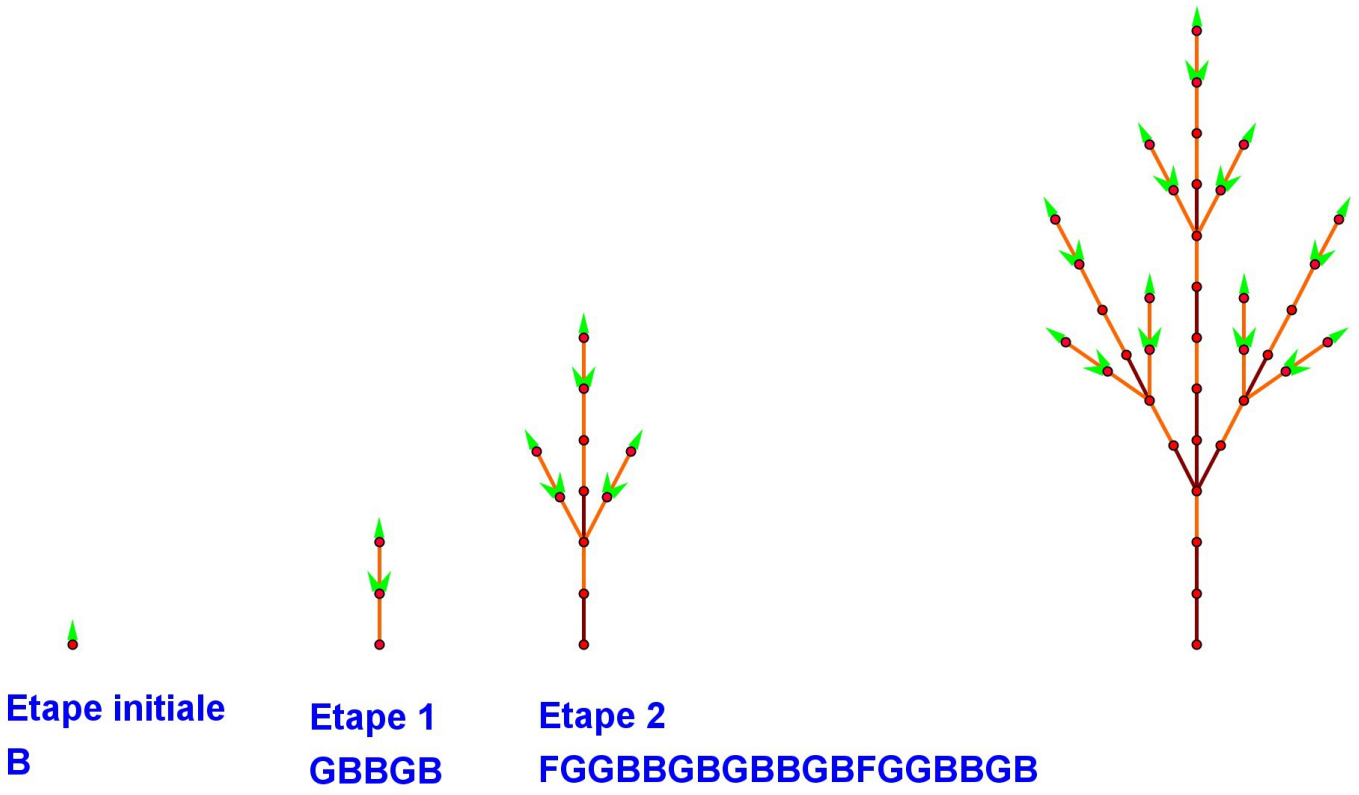


Illustration 2: Évolutions du nouveau modèle

Avec n=0 à l'étape initiale on obtient : B_0=1, F_0=0 et G_0=0

Nombre de bourgeons $B_n=3^n$ comme précédemment nous avons un suite géométrique de raison 3 et de premier terme $B_0=1$.

Nombre de tiges G $G_n=3^n-1$

Nombre de tiges F $F_n=\frac{1}{2}(3^n-1)-n$

Nombre de tiges total $T_n=\frac{3}{2}(3^n-1)-n$

Comment avons nous trouvé ces formules ?

$G_{n+1}=2\times 3^n+G_n$

$G_{n+2}=3^{n+1}\times 2+G_{n+1}=3^{n+1}\times 2+3^n\times 2+G_n=(3^1+3^0)(3^n\times 2)+G_n$

$G_{n+3}=3^{n+2}\times 2+G_{n+2}=3^{n+2}\times 2+(3^1+3^0)(3^{n+1}\times 2)+G_n=(3^2+3^1+3^0)(3^n\times 2)+G_n$

Que l'on généralise $G_{n+p}=(3^{p-1}+3^p+...+3^0)(3^n\times 2)+G_n$ en prenant $n=0$ et sachant que $G_0=0$, on obtient : $G_p=2\times(3^{p-1}+...+3^1+3^0)=2\sum_{p=0}^{p-1}3^p=3^p-1$

Pour n⩾1 nous avons $F_n=G_{n-1}+F_{n-1}$ et $F_0=0$

$F_n=G_{n-1}+G_{n-2}+F_{n-2}=\sum_{k=1}^{n}G_{n-k}=\sum_{k=1}^{n}(3^{n-k}-1)=\sum_{k=0}^{n-1}3^k-n=\frac{3^n-1}{3-1}-n=\frac{1}{2}(3^n-1)-n$

et $T_n=G_n+F_n=3^n-1+\frac{1}{2}(3^n-1)-n=\frac{3}{2}(3^n-1)-n$

IV. Généralisation des deux modèles

Nos collègues roumains ont travaillé sur la généralisation des deux modèles.

F → (n_1 fois)F

B → (n_2 fois)F[+B][-B](n_3 fois)FB

où n_1, n_2 et n_3 sont des entiers (dans l'exemple précédent $n_1=2$, $n_2=n_3=1$)

Nous n'allons pas regarder l'aspect graphique mais seulement le nombre de B et de F.

Comme précédemment nous avons $B_n=3^n$ en effet la suite B_n est géométrique de raison 3.

Nous avons $F_{n+1}=n_1\times F_n+(n_2+n_3)\times B_n=n_1\times F_n+3^n(n_2+n_3)$

En appliquant la même méthode, nous avons :

$F_{n+2}=n_1\times F_{n+1}+3^{n+1}\times(n_2+n_3)=n_1\times(n_1\times F_n+3^n\times(n_2+n_3))+3^{n+1}\times(n_2+n_3)$

$F_{n+2}=n_1\times(n_1\times F_n+3^n\times(n_2+n_3))+3^{n+1}\times(n_2+n_3)=n_1^2F_n+n_1\times 3^n\times(n_2+n_3)+3^{n+1}\times(n_2+n_3)$

$F_{n+2}=n_1^2F_n+3^n(n_2+n_3)(3+n_1)$

$F_{n+3}=n_1\times F_{n+2}+3^{n+2}\times(n_2+n_3)=n_1(n_1^2F_n+3^n(n_2+n_3)(3+n_1))+3^{n+2}(n_2+n_3)$

$F_{n+3}=n_1^3F_n+n_1 3^n(n_2+n_3)(3+n_1)+3^{n+2}(n_2+n_3)=n_1^3F_n+3^n(n_2+n_3)[3n_1+n_1^2]+3^n\times 3^2(n_2+n_3)$

$F_{n+3}=n_1^3F_n+3^n(n_2+n_3)(3n_1+n_1^2+3^2)$

$F_{n+4}=n_1\times F_{n+3}+3^{n+3}\times(n_2+n_3)=n_1(n_1^3F_n+3^n(n_2+n_3)(3n_1+n_1^2+3^2))+3^{n+3}\times(n_2+n_3)$

$F_{n+4}=n_1^4F_n+n_1 3^n(n_2+n_3)(3n_1+n_1^2+3^2)+3^{n+3}\times(n_2+n_3)=n_1^4F_n+3^n(n_2+n_3)(3n_1^2+n_1^3+3^2n_1)+3^n\times 3^3\times(n_2+n_3)$

$F_{n+4}=n_1^4F_n+3^n(n_2+n_3)(3n_1^2+n_1^3+3^2n_1+3^3)$

Si on généralise : $F_{n+p}=n_1^p F_n+3^n(n_2+n_3)(3^{p-1}n_1^0+3^{p-2}n_1^1+3^{p-3}n_1^2+\ldots+3^0 n_1^{p-1})$ avec $p\geqslant 1$.
Si on prend n=0 : $F_p=(n_2+n_3)(3^{p-1}n_1^0+3^{p-2}n_1^1+3^{p-3}n_1^2+\ldots+3^0 n_1^{p-1})$

Si n_1=3, $F_p=(n_2+n_3)(3^{p-1}3^0+3^{p-2}3^1+3^{p-3}3^2+\ldots+3^0 3^{p-1})=p(n_2+n_3)\times 3^{p-1}$
Si $n_1\neq 3$, nous appliquons l'astuce roumaine de diviser par n_1^{p-1}

$$\frac{F_p}{n_1^{p-1}}=\frac{(n_2+n_3)(3^{p-1}n_1^0+3^{p-2}n_1^1+3^{p-3}n_1^2+\ldots+3^0 n_1^{p-1})}{n_1^{p-1}}=(n_2+n_3)\left(\frac{3^{p-1}n_1^0}{n_1^{p-1}}+\frac{3^{p-2}n_1^1}{n_1^{p-1}}+\frac{3^{p-3}n_1^2}{n_1^{p-1}}+\ldots+\frac{3^0 n_1^{p-1}}{n_1^{p-1}}\right)$$

$$\frac{F_p}{n_1^{p-1}}=(n_2+n_3)\left(\frac{3^{p-1}}{n_1^{p-1}}+\frac{3^{p-2}}{n_1^{p-2}}+\frac{3^{p-3}}{n_1^{p-3}}+\ldots+\frac{3^0}{n_1^0}\right)=(n_2+n_3)\frac{\left(\frac{3}{n_1}\right)^p-1}{\frac{3}{n_1}-1}=(n_2+n_3)\frac{3^p-n_1^p}{3-n_1}\times\frac{1}{n_1^{p-1}}$$

ce qui donne $F_p=(n_2+n_3)\frac{3^p-n_1^p}{3-n_1}$
Remarque : pour n_1=2, n_2=1 et n_3=1 on retrouve le résultat de la propriété 3.

La généralisation du deuxième modèle donne :
F → F
G → F(n_1)G
B → (n_2)G[+B][-B](n_3)GB
Nous conduit à :
$B_0=1$ et $B_{n+1}=3B_n$ donc $B_n=3^n$

$G_0=0$ et $G_{n+1}=n_1\times G_n+(n_2+n_3)\times 3^n$ qui donne (d'après le résultat précédent) $G_n=(n_2+n_3)\frac{3^p-n_1^p}{3-n_1}$ si $n_1\neq 3$

$F_0=0$ et $F_{n+1}=F_n+G_n=\sum_{k=0}^{n} G_k$

Rédaction des travaux de recherche

Annexe 1 – Programme qui détermine le nombre de B et de F

```
#include <iostream>
#include <string.h>
using namespace std;
int putere(int n, int p)
{
  int i,nr;
  i=1;
  nr=n;
  while(i<p)
  {
  n=nr*n;
  i++;
  }
  if(p==0)
  return 1;
  return n;
}
int suma(int n)
{
  int i,s=0;
  i=1;
  while(i<=n)
  {
s=s+putere(2,n-i);
  i++;
  }
  return s;
}
int main()
{
  char x[1000000]="B",a,c[1000000],d[100];
  int i,n,j,b,k,q,w,r,u,lng,f=0,bo;
  k=0;
  j=0;
cout<<"Introduisez le nombre des etapes de la croissance de la fougere: ";
  cin>>n;
  cout<<endl;
strcpy(c,x);
  i=0;
  lng=0;
  if(i==0)
  {
cout<<"Etat initiale"<<": "<<x<<endl;
bo=putere(3,i);
cout<<"Il y a "<<bo<<" B"<<endl<<endl;
  }
for(i=1;i<=n;i++)
  {
lng=lng+putere(2,i);
cout<<"Etape "<<i<<": ";
b=putere(2,i-1);
while(b!=0)
  {
for(j=strlen(x)-1;j>=0;j--)
  {
x[j+1]=x[j];
  }
x[j+1]='F';
  b--;
  }
  q=1;
j=strlen(x);
while(q!=0)
  {
for(w=0;w<=strlen(c)-1;w++)
  {
x[j++]=c[w];
  }
  q--;
  }
x[j]=0;
b=putere(2,i-1);
j=strlen(x);
while(b!=0)
  {
x[j++]='F';
  b--;
  }
x[j]=0;
j=strlen(x);
for(w=0;w<=strlen(c)-1;w++)
  {
x[j++]=c[w];
  }
x[j]=0;
cout<<x;
cout<<endl;
cout<<"La longeur de la branche centrale est: "<<lng<<endl;
strcpy(c,x);
f=2*putere(3,i-1)+2*f;
bo=putere(3,i);
cout<<"Il y a "<<f<<" F et "<<bo<<" B"<<endl<<endl;
  }
  return 0;
}
```

Annexe 2 – Démonstration de la propriété 3

Lors des échanges avec notre chercheur, ce dernier, avec l'aide du logiciel Mathematica nous a proposé une formule pour F_n.

```
In[5]:= f[n_] := Sum[2^k × 3^(n - k), {k, 1, n}]
        Map[f, Range[16]]
Out[7]= {2, 10, 38, 130, 422, 1330, 4118, 12 610, 38 342,
         116 050, 350 198, 1 054 690, 3 172 262, 9 533 170, 28 632 278, 85 962 370}

In[8]:= f[n]
Out[8]= -2 (2^n - 3^n)
```

Démonstration par récurrence que $F_n = 2\times\sum_{k=0}^{n-1} 2^k\times 3^{n-1-k} = 2(3^n - 2^n)$ pour n≥1.

Initialisation : pour n=1

$2(3^{n-1}\times 2^0 + \ldots + 2^{n-1}\times 3^0) = 2(3^0\times 2^0) = 2$

$2(3^n - 2^n) = 2$

donc pour n=1 on a $F_n = 2(3^{n-1}\times 2^0 + \ldots + 2^{n-1}\times 3^0) = 2(3^n - 2^n)$

Hérédité : On suppose qu'il existe un n tel que $2(3^{n-1}\times 2^0 + \ldots + 2^{n-1}\times 3^0) = 2(3^n - 2^n)$

On veut démontrer que $2(3^n\times 2^0 + \ldots + 2^n\times 3^0) = 2(3^{n+1} - 2^{n+1})$

$2(3^n\times 2^0 + 3^{n-1}\times 2^1 + \ldots + 2^{n-1}\times 3^1 + 2^n\times 3^0) = 2(3^n\times 2^0 + 2(3^{n-1}\times 2^0 + 3^{n-2}\times 2^1 + \ldots + 3^0\times 2^{n-1}))$

or $2(3^{n-1}\times 2^0 + \ldots + 2^{n-1}\times 3^0) = 2(3^n - 2^n)$ Hypothèse de récurrence

donc $2(3^n\times 2^0 + 3^{n-1}\times 2^1 + \ldots + 2^{n-1}\times 3^1 + 2^n\times 3^0) = 2(3^n\times 1 + 2(3^n - 2^n)) = 2(3^n + 2\times 3^n - 2\times 2^n)$

$F_{n+1} = 2(3\times 3^n - 2^{n+1})$

$F_{n+1} = 2(3^{n+1} - 2^{n+1})$

Conclusion : quel que soit n≥1 on a $F_n = 2(3^{n-1}\times 2^0 + \ldots + 2^{n-1}\times 3^0) = 2(3^n - 2^n)$

Remarque : la formule $F_n = 2(3^n - 2^n)$ marche aussi pour n=0.

Démonstration de la propriété 3 en partant de la formule (3) par les élèves de Cluj :

Formule (3) $F_n = 3F_{n-1} + 2^n$

$F_{n-1} = 3F_{n-2} + 2^{n-1}$ $\times 3$

$F_{n-2} = 3F_{n-3} + 2^{n-2}$ $\times 3^2$

…

$F_2 = 3F_1 + 2^2$ $\times 3^{n-2}$

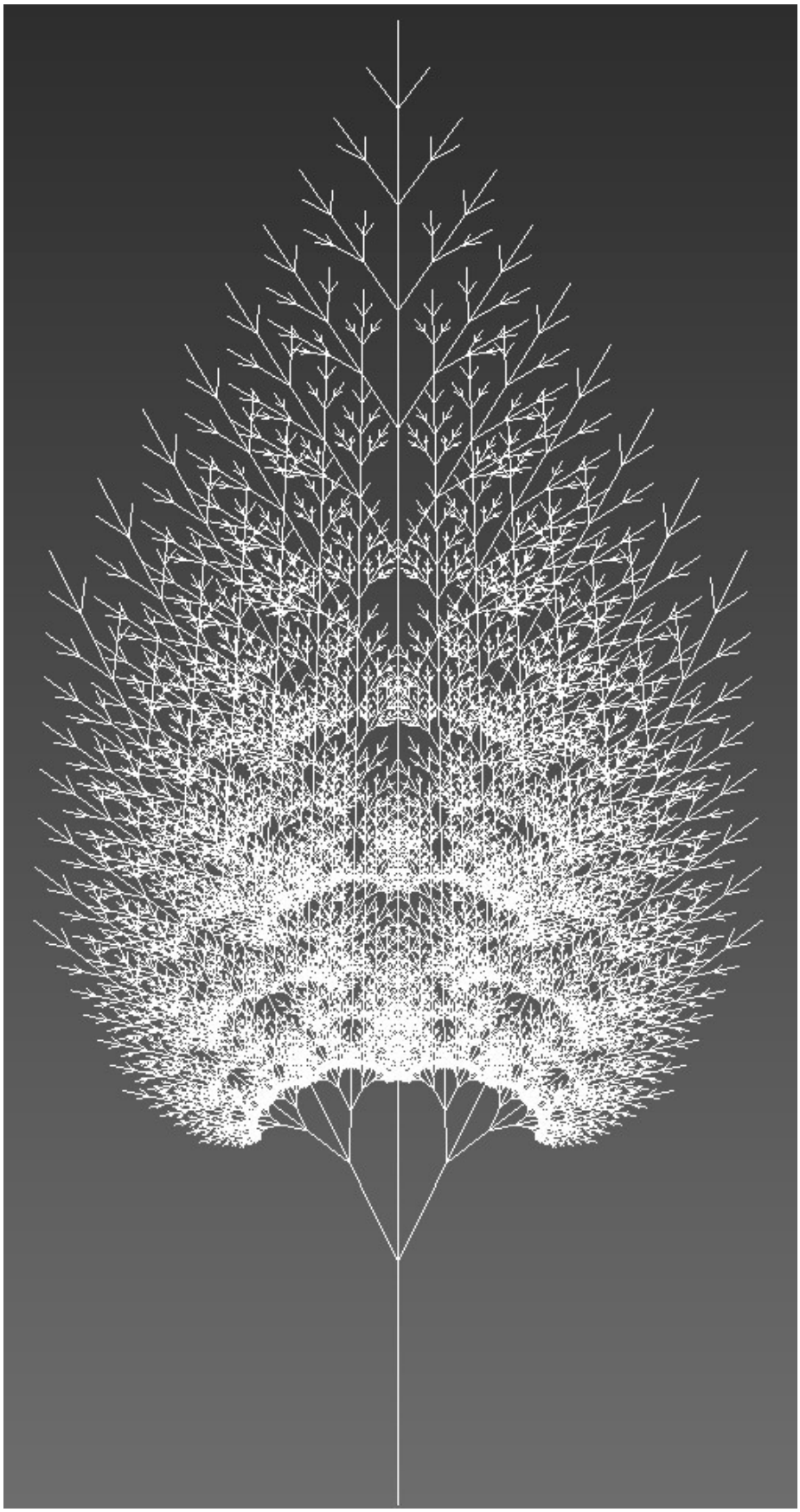
$F_1 = 3F_0 + 2$ $\times 3^{n-1}$

Si l'on fait la somme, de nombreux termes se simplifies :

$$F_n + 3F_{n-1} + 3^2F_{n-2} + \ldots + 3^{n-2}F_2 + 3^{n-1}F_1 = 3F_{n-1} + 3\times 3F_{n-2} + 3\times 3^2F_{n-3} + \ldots + 3\times 3^{n-2}F_1 + 3^nF_0 + \sum_{k=1}^{n} 2^k 3^{n-k}$$

Sachant que $F_0 = 0$, on obtient : $F_n = \sum_{k=1}^{n} 2^k\times 3^{n-k}$

Annexe 3 – programme des élèves roumains en C++



Résultat de la dixième génération. Le temps de calculs fait que le programme ne peut pas aller plus loin

main.cpp | scene.wrl

```cpp
#include <iostream>
#include <cmath>
#include <fstream>
using namespace std;
#define PI 3.141592
ofstream fo ("scene.wrl");
typedef struct punct {double x, y, z;} PUNCT;
void line (PUNCT A, PUNCT B)
{
    fo<<"Shape{\n";
    fo<<"geometry IndexedLineSet{\n";
    fo<<"coord Coordinate{\n";
    fo<<"point [\n";
    fo<<A.x<<" "<<A.y<<" "<<A.z<<",\n";
    fo<<B.x<<" "<<B.y<<" "<<B.z<<",\n";
    fo<<"]\n";
    fo<<"}\n";
    fo<<"coordIndex [\n";
    fo<<"0, 1, -1,\n";
    fo<<"]\n";
    fo<<"}\n";
    fo<<"}\n";
}
void fern_creation (PUNCT P, int n, double l, double u)
{
    PUNCT P1;
    if(n>0)
    {
        P1.z=P.z;
        P1.x=P.x+l*cos(u);
        P1.y=P.y+l*sin(u);
        line(P, P1);
        fern_creation (P1, n-1, l/2.0, u-PI/7);
        fern_creation (P1, n-1, l/2.0, u+PI/7);
        fern_creation (P1, n-1, l, u);
    }
}
PUNCT P;

int main()
{
    fo<<"#VRML V2.0 utf8\n";
    P.x=0;
    P.y=0;
    P.z=0;
    fern_creation (P, 7, 2, PI/2);
    fo.close();
    return 0;
}
```

GLOSSAIRE

Sujet : la fougère

Tige (stem) : dans ce texte, une tige peut être vue comme un segment

Bourgeon (bad) : dans ce texte, un bourgeon peut être vu comme un tout petit vecteur

Fougère (ferm) : ce modèle de L-système conduit à une représentation qui ressemble à une fougère

Suite (sequence) : fonction dont le domaine de définition est $\mathbb{N}$. Dans ce cas on note la variable n et la suite u s'écrit $u(n)=u_n$.

Suite géométrique (geometrical sequence) : une suite est dite géométrique de raison $q \neq 1$ (common ratio) si pour tout entier n, $u_{n+1}=q \times u_n$

Démonstration par récurrence (proof by induction) : procédé de démonstration propre aux propriétés liées à une variable n entière.